

本节内容

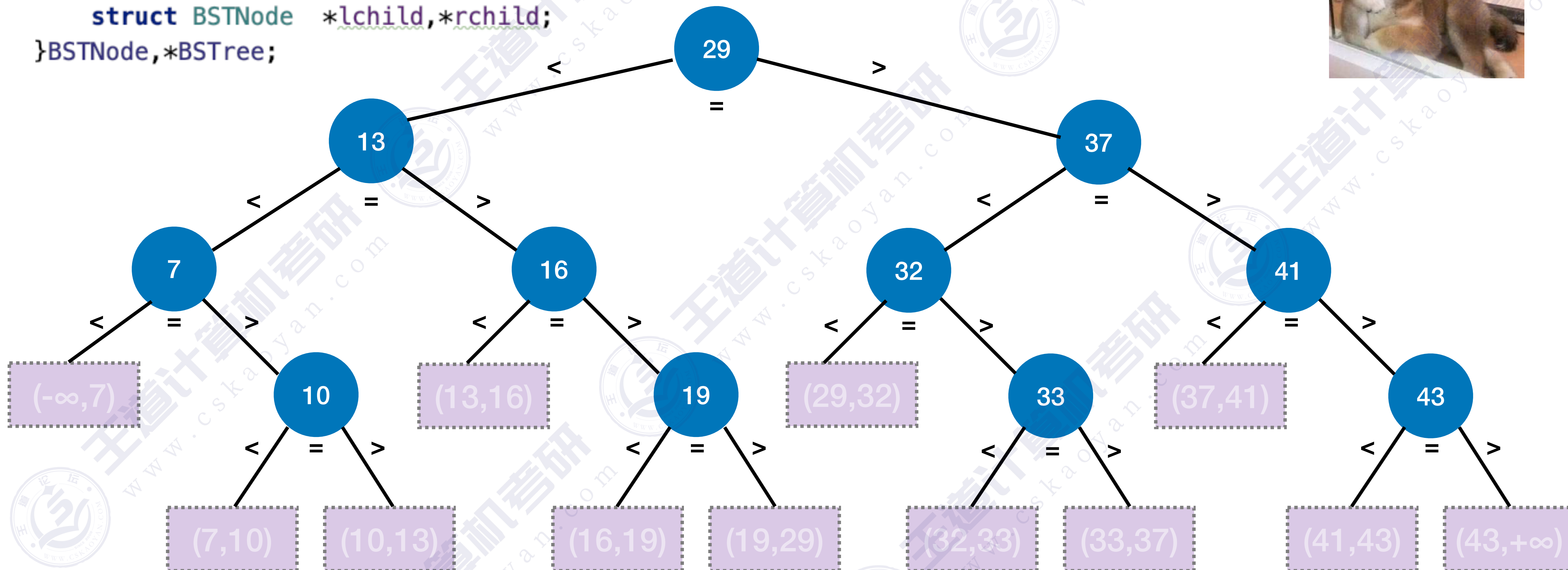
B树

回顾：二叉查找树 (BST)

// 二叉排序树结点

```
typedef struct BSTNode{  
    int key;  
    struct BSTNode *lchild,*rchild;  
}BSTNode,*BSTree;
```

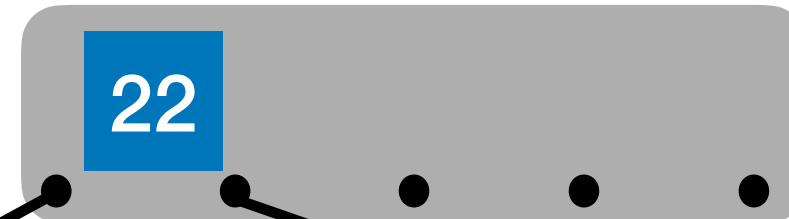
能不能变成m
叉查找树?



5叉查找树

//5叉排序树的结点定义

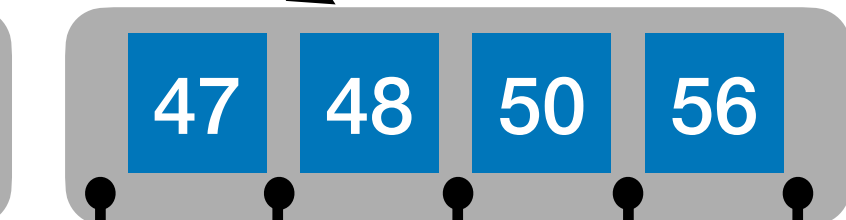
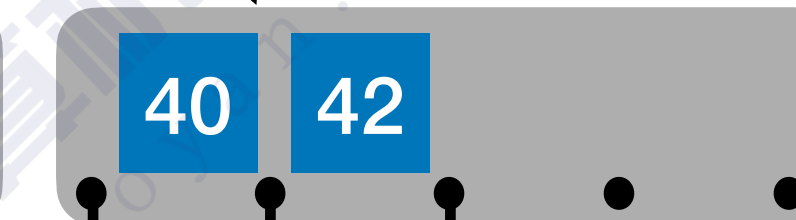
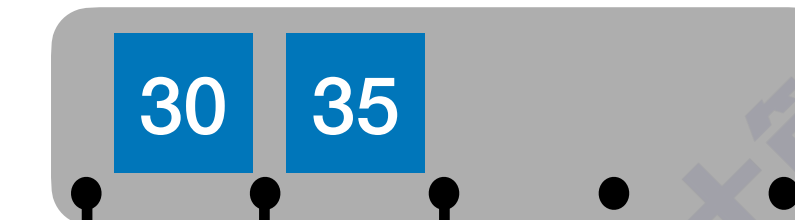
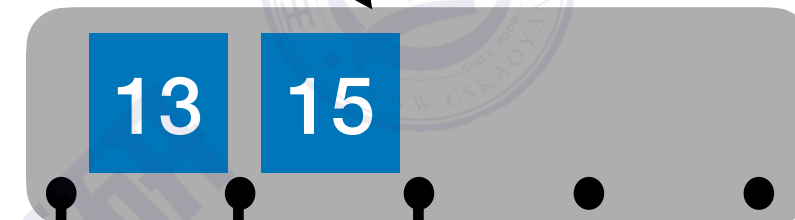
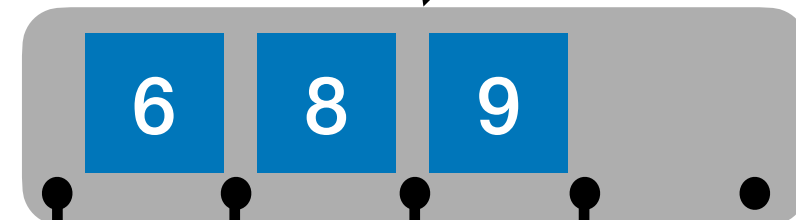
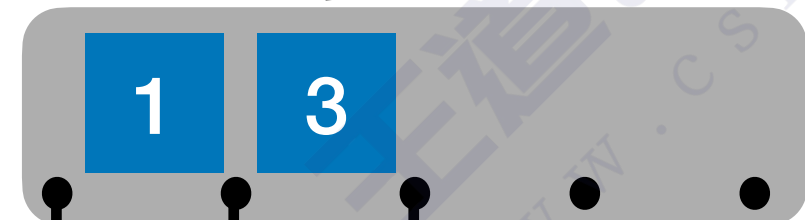
```
struct Node {  
    ElemType keys[4];           //最多4个关键字  
    struct Node * child[5];    //最多5个孩子  
    int num;                   //结点中有几个关键字  
};
```



最少1个关键字，2个分叉
最多4个关键字，5个分叉



结点内关键字有序

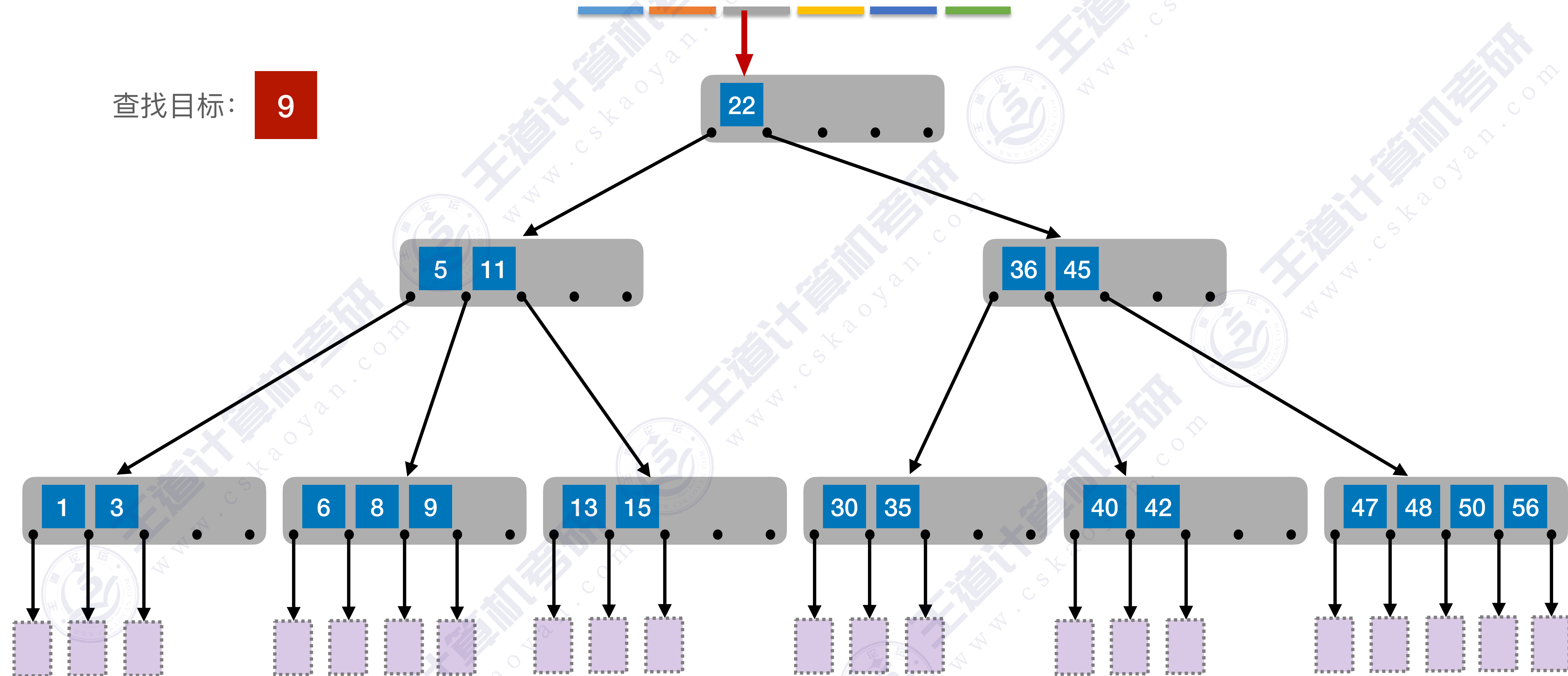


失败结点

如何查找

查找目标：

9

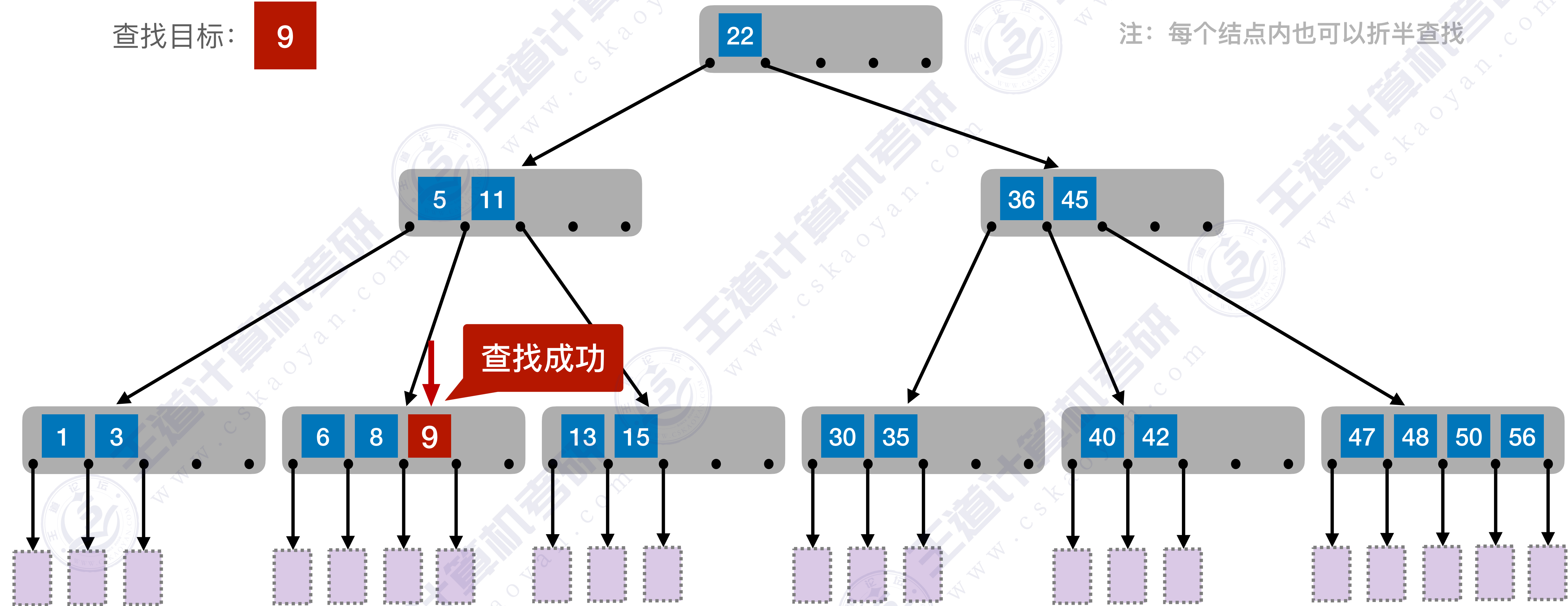


如何查找

查找目标：

9

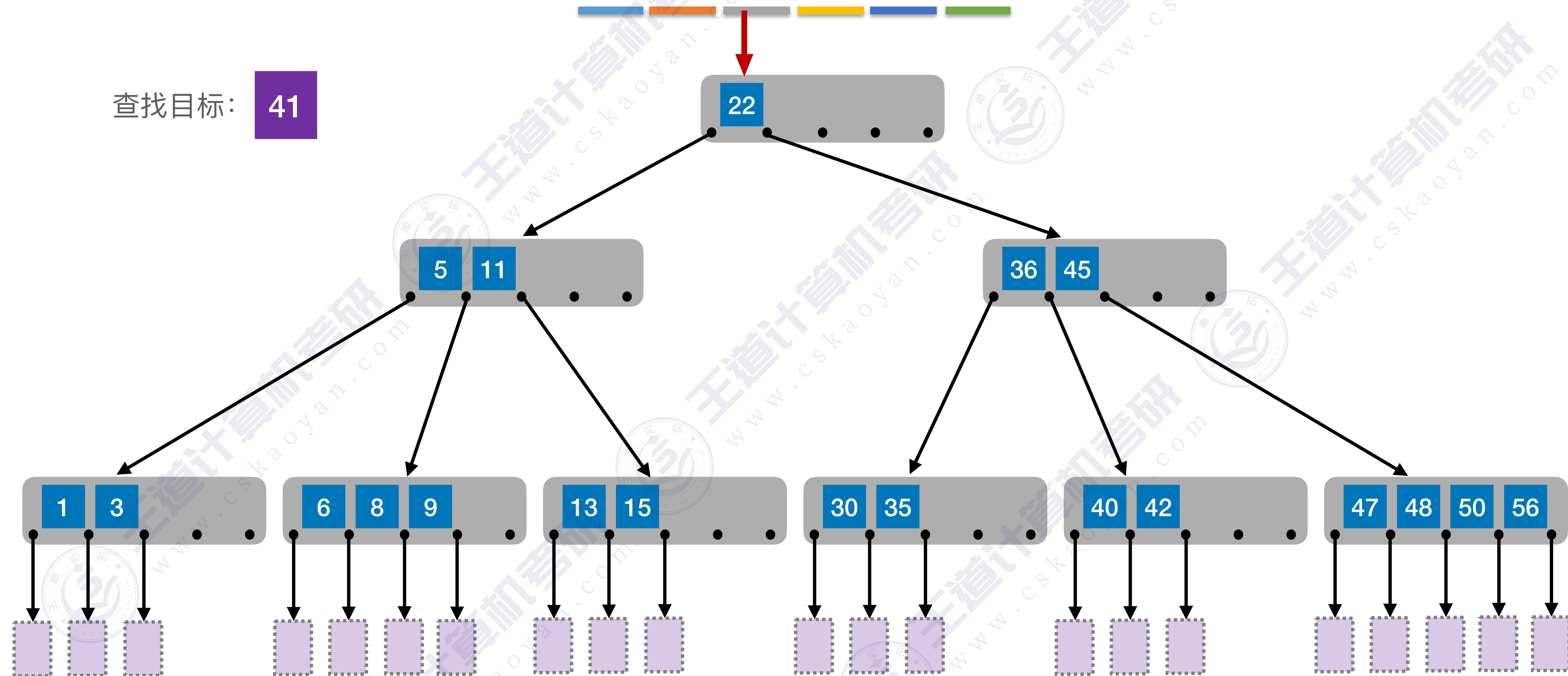
注：每个结点内也可以折半查找



如何查找

查找目标：

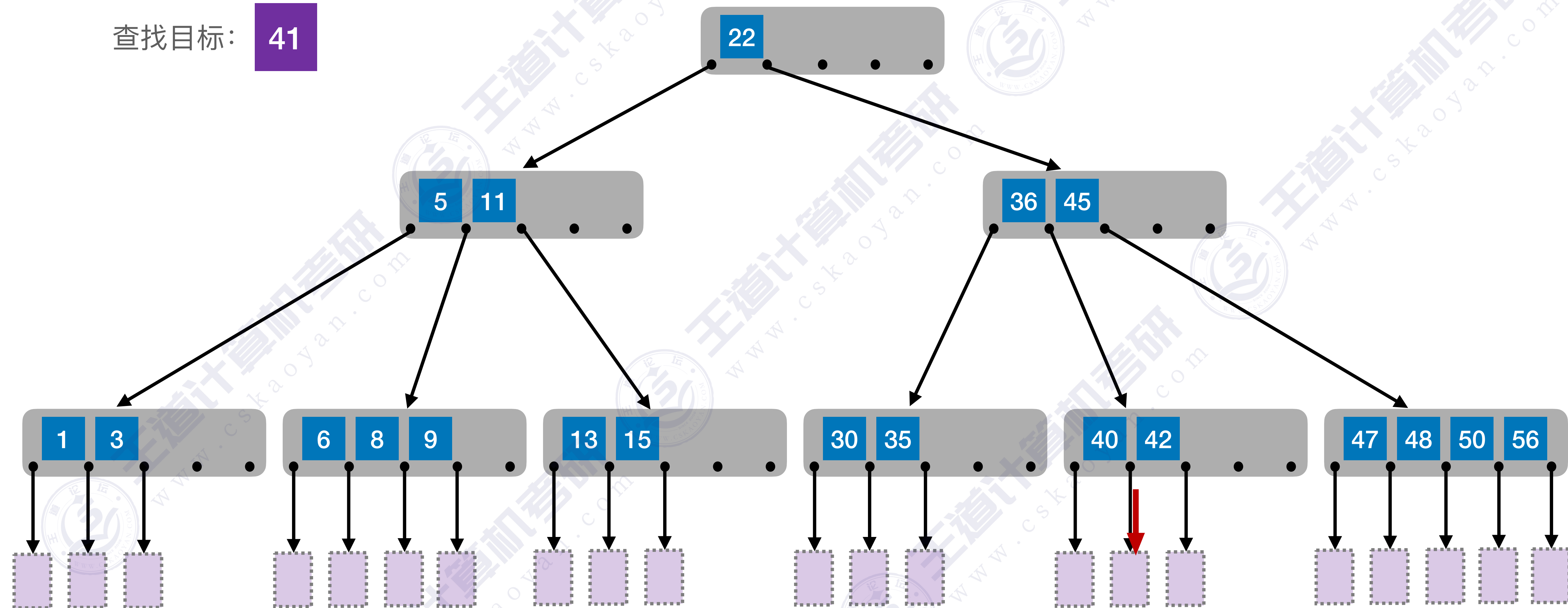
41



如何查找

查找目标：

41

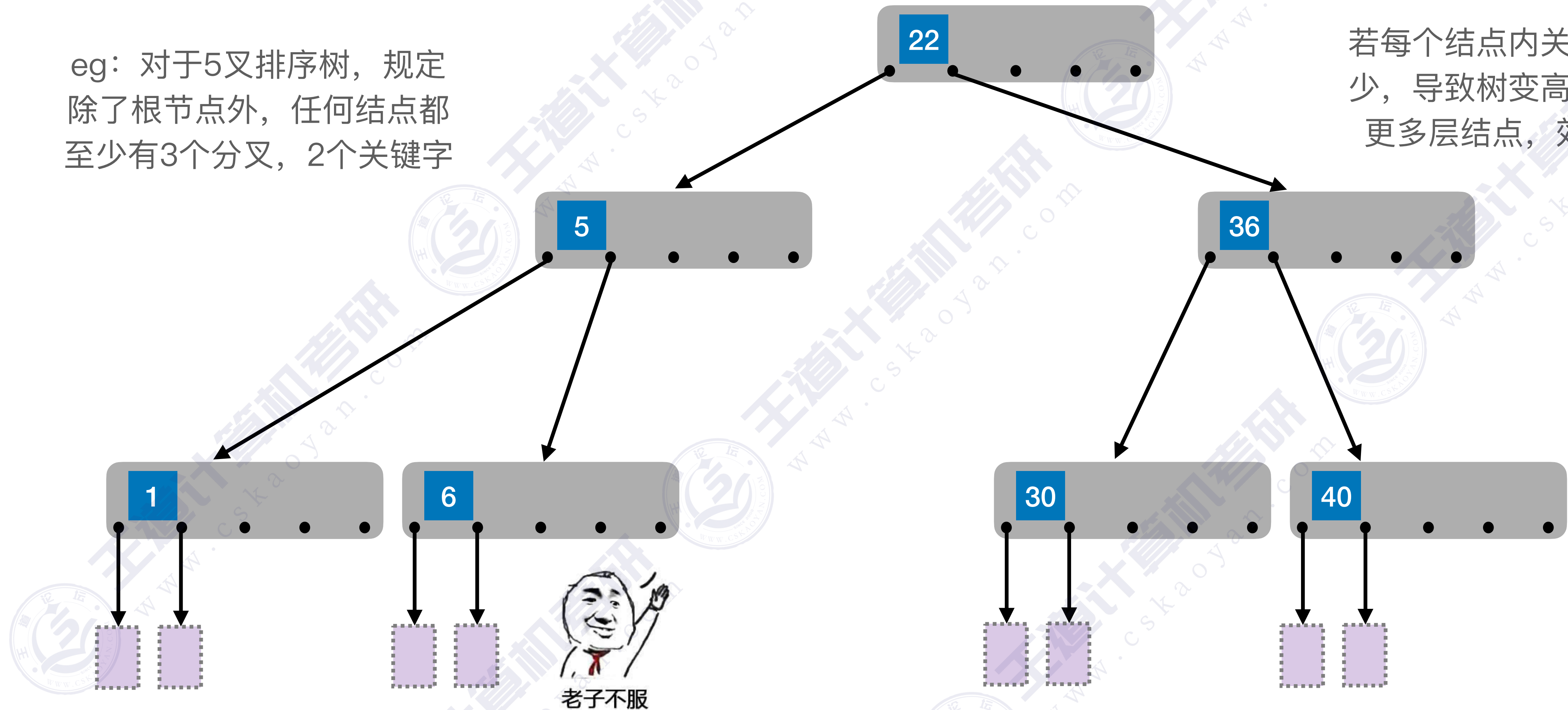


查找失败

如何保证查找效率

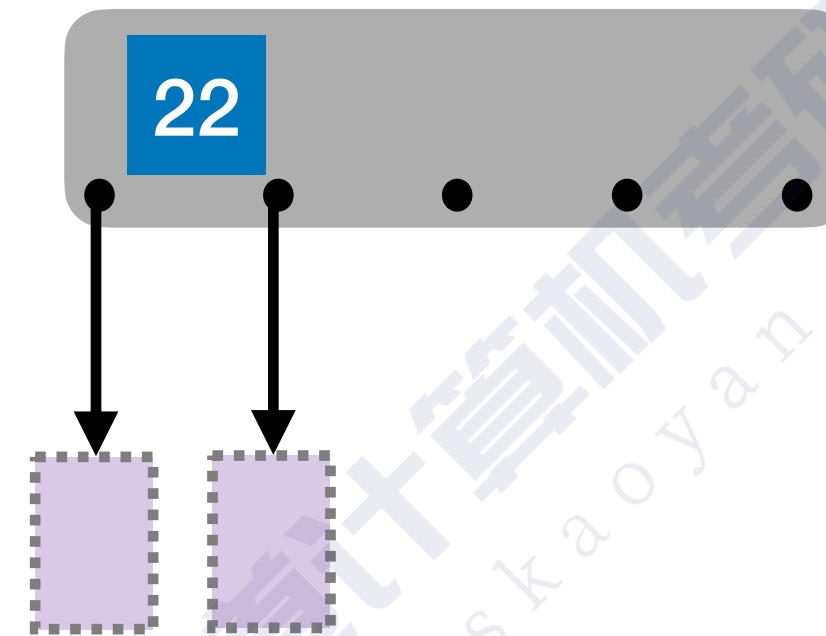
eg: 对于5叉排序树, 规定除了根节点外, 任何结点都至少有3个分叉, 2个关键字

若每个结点内关键字太少, 导致树变高, 要查更多层结点, 效率低



策略: m 叉查找树中, 规定除了根节点外, 任何结点至少有 $\lceil m/2 \rceil$ 个分叉, 即至少含有 $\lceil m/2 \rceil - 1$ 个关键字

如何保证查找效率



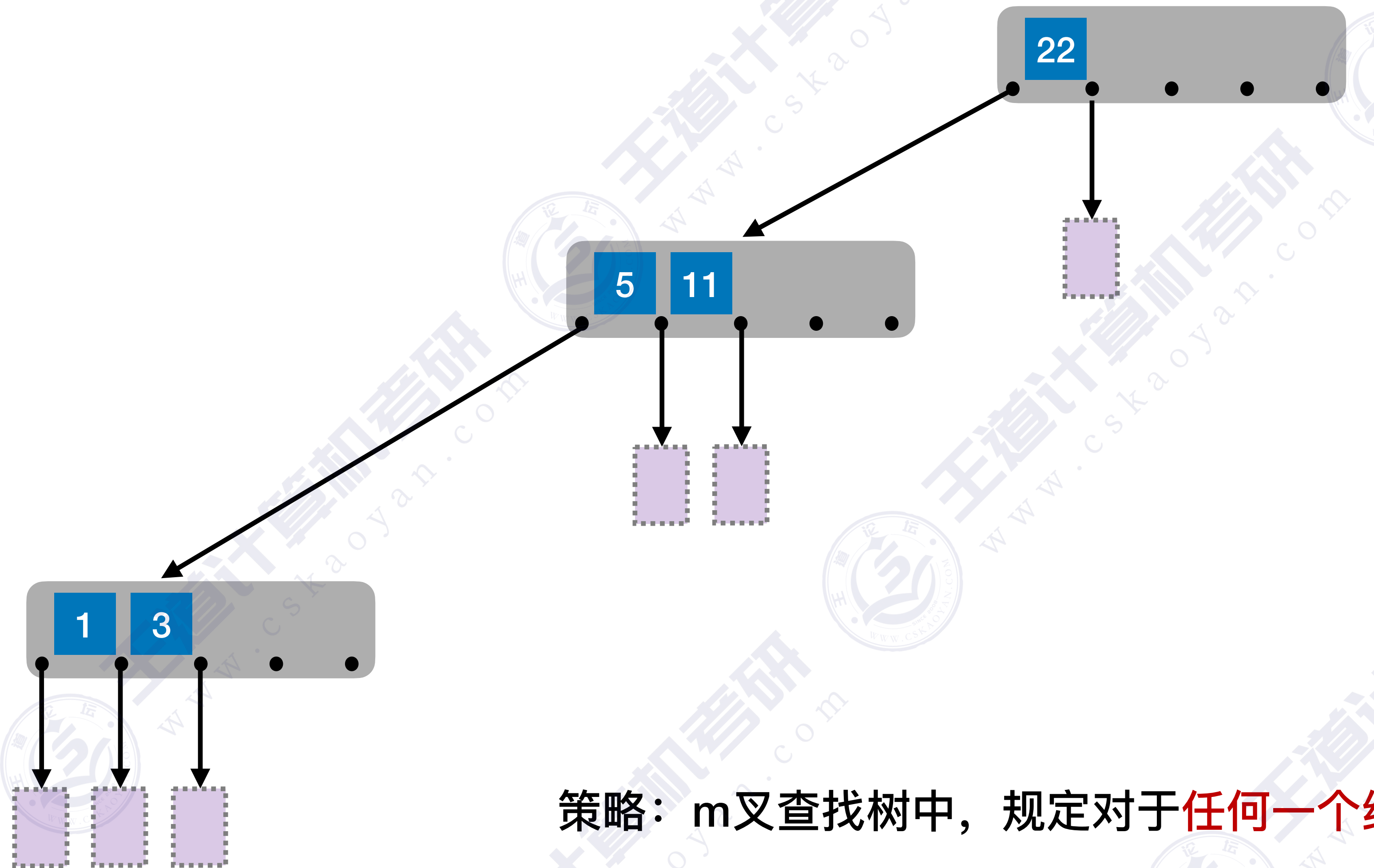
这个我也没有 没办法



如果整个树只有1个元素，根节点只有两个分叉

策略：m叉查找树中，规定除了根节点外，任何结点至少有 $\lceil m/2 \rceil$ 个分叉，即至少含有 $\lceil m/2 \rceil - 1$ 个关键字

如何保证查找效率

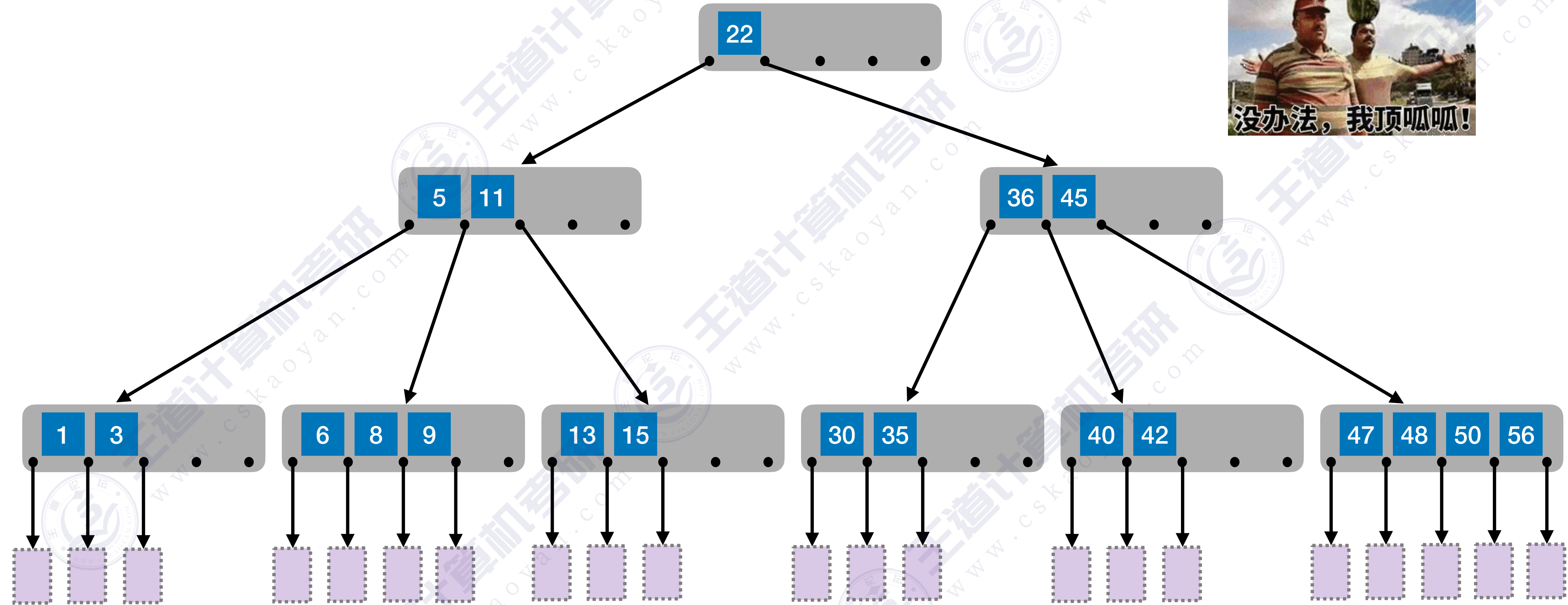


还不够

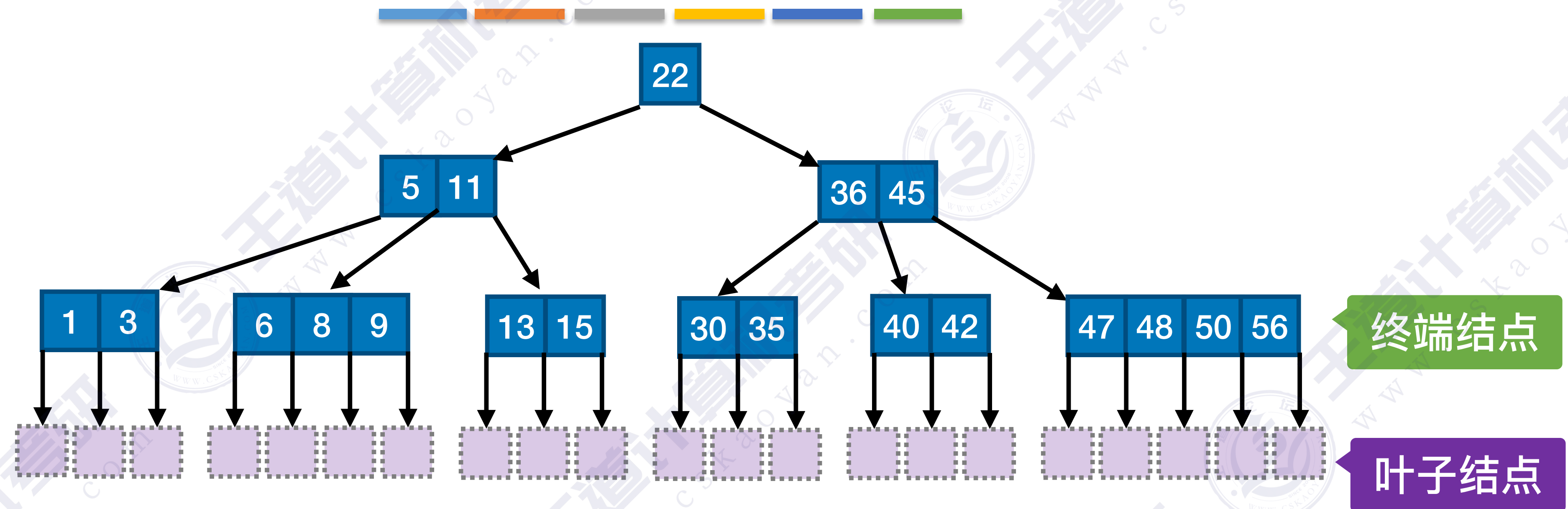
不够“平衡”，树会很高，要查很多层结点

策略：m叉查找树中，规定对于**任何一个结点**，其所有子树的高度都要相同。

B树



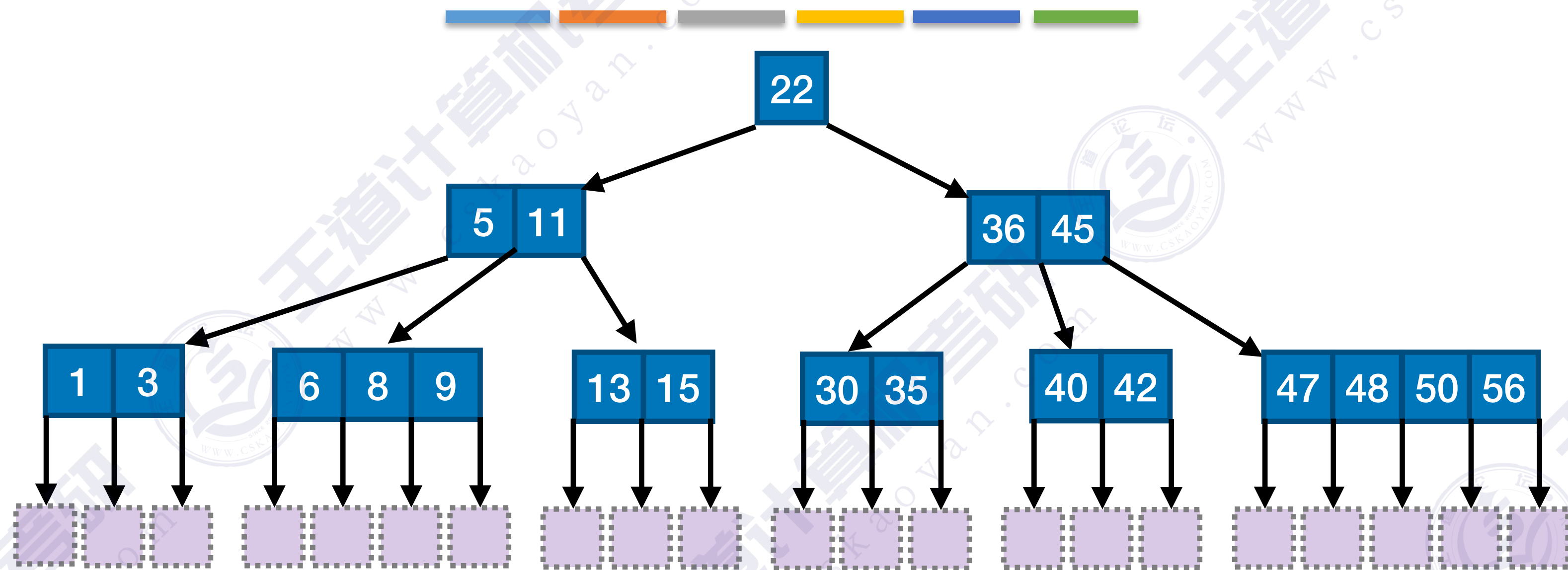
B树



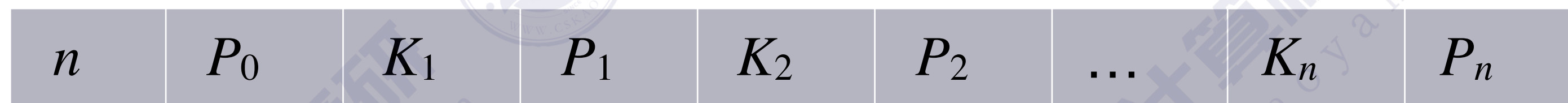
B树，又称**多路平衡查找树**，B树中所被允许的孩子个数的最大值称为B树的阶，通常用 m 表示。一棵 **m 阶B树**或为空树，或为满足如下特性的 m 叉树：

- 1) 树中每个结点至多有 m 棵子树，即至多含有 $m-1$ 个关键字。
- 2) 若根结点不是终端结点，则至少有两棵子树。
- 3) **除根结点外的所有非叶结点至少有 $\lceil m/2 \rceil$ 棵子树**，即至少含有 $\lceil m/2 \rceil - 1$ 个关键字。
- 5) 所有的**叶结点**都出现在同一层次上，并且不带信息（可以视为外部结点或类似于折半查找判定树的查找失败结点，实际上这些结点不存在，指向这些结点的指针为空）。

B树



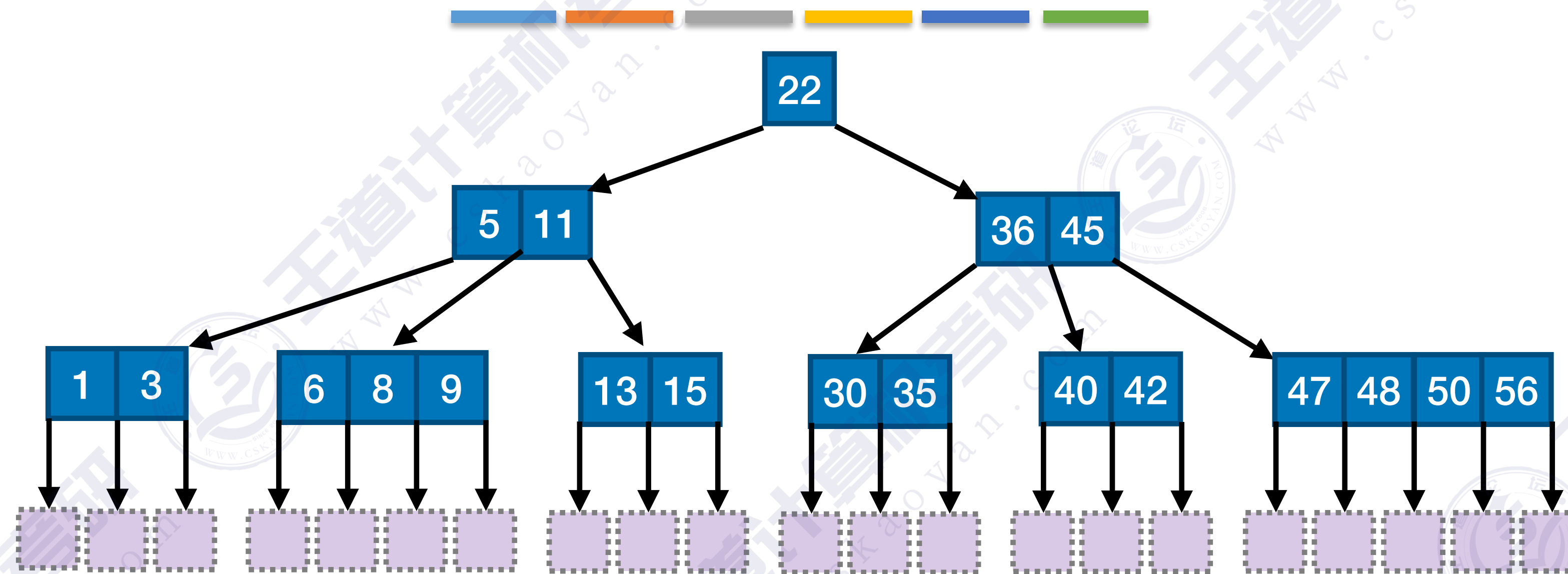
4) 所有非叶结点的结构如下:



其中, K_i ($i = 1, 2, \dots, n$) 为结点的关键字, 且满足 $K_1 < K_2 < \dots < K_n$; P_i ($i = 0, 1, \dots, n$) 为指向子树根结点的指针, 且指针 P_{i-1} 所指子树中所有结点的关键字均小于 K_i , P_i 所指子树中所有结点的关键字均大于 K_i , n

($\lceil m/2 \rceil - 1 \leq n \leq m - 1$) 为结点中关键字的个数。

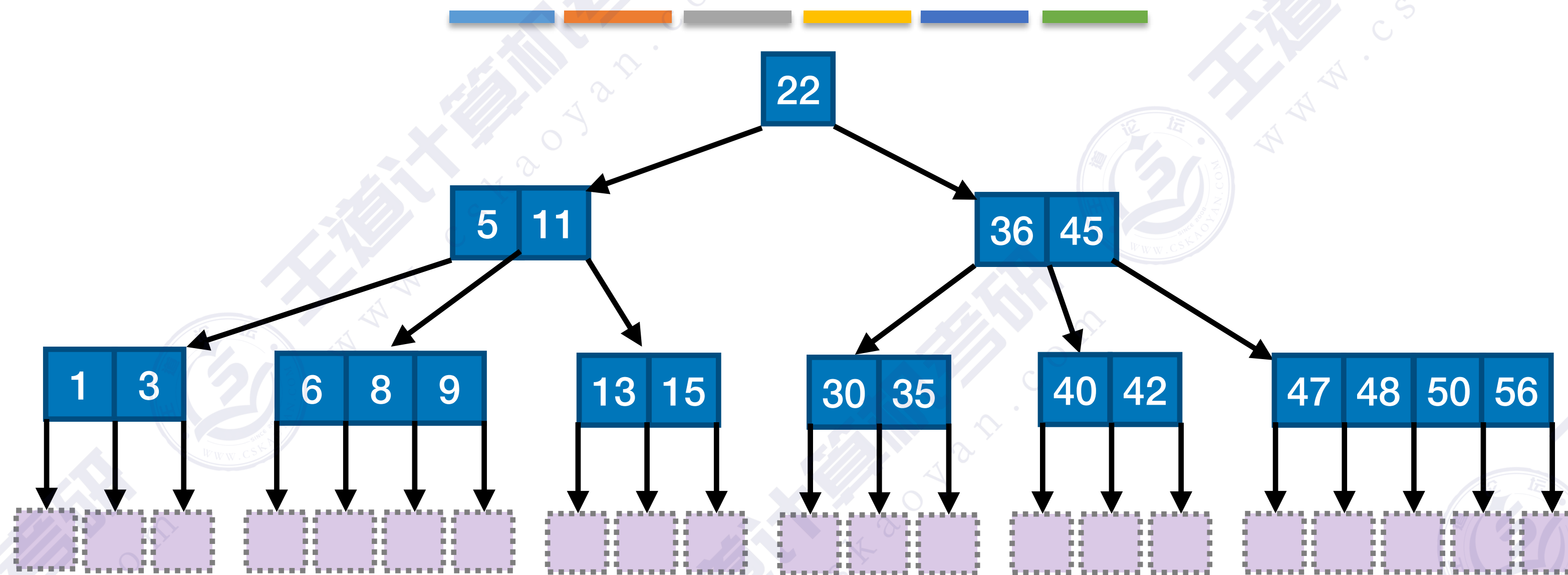
B树



m阶B树的核心特性:

- 1) 根节点的子树数 $\in [2, m]$, 关键字数 $\in [1, m-1]$ 。
其他结点的子树数 $\in [\lceil m/2 \rceil, m]$; 关键字数 $\in [\lceil m/2 \rceil - 1, m-1]$
- 2) 对任一结点, 其所有子树高度都相同
- 3) 关键字的值: 子树0 < 关键字1 < 子树1 < 关键字2 < 子树2 < (类比二叉查找树 左 < 中 < 右)

B树的高度

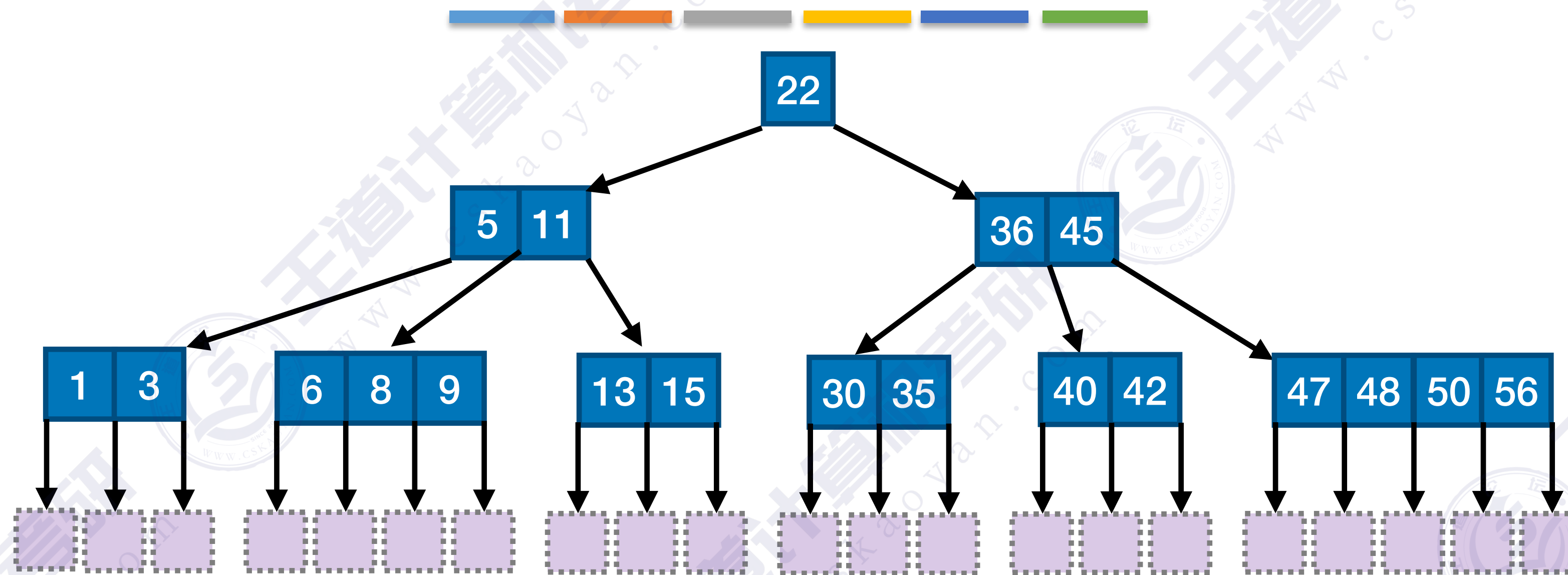


注：大部分学校算B树的**高度不包括叶子结点**（失败结点）

问题：含n个关键字的m阶B树，最小高度、最大高度是多少？

最小高度——让每个结点尽可能的满，有m-1个关键字，m个分叉，则有
$$n \leq (m-1)(1 + m + m^2 + m^3 + \dots + m^{h-1}) = m^h - 1$$
，因此 $h \geq \log_m(n+1)$

B树的高度



注：大部分学校算B树的**高度不包括叶子结点（失败结点）**

最大高度——让各层的分叉尽可能的少，即根节点只有2个分叉，其他结点只有 $\lceil m/2 \rceil$ 个分叉
各层结点至少有：第一层 1、第二层 2、第三层 $2\lceil m/2 \rceil$... 第h层 $2(\lceil m/2 \rceil)^{h-2}$
第h+1层共有**叶子结点（失败结点）** $2(\lceil m/2 \rceil)^{h-1}$ 个

n个关键字的B树必有n+1个叶子结点，则 $n + 1 \geq 2(\lceil m/2 \rceil)^{h-1}$ ，即 $h \leq \log_{\lceil m/2 \rceil} \frac{n+1}{2} + 1$

n个关键字将数域切分为n+1个区间

B树的高度

问题：含n个关键字的m叉B树，最小高度、最大高度是多少？

最大高度——让每个结点包含的关键字、分叉尽可能的少。记 $k=\lceil m/2 \rceil$

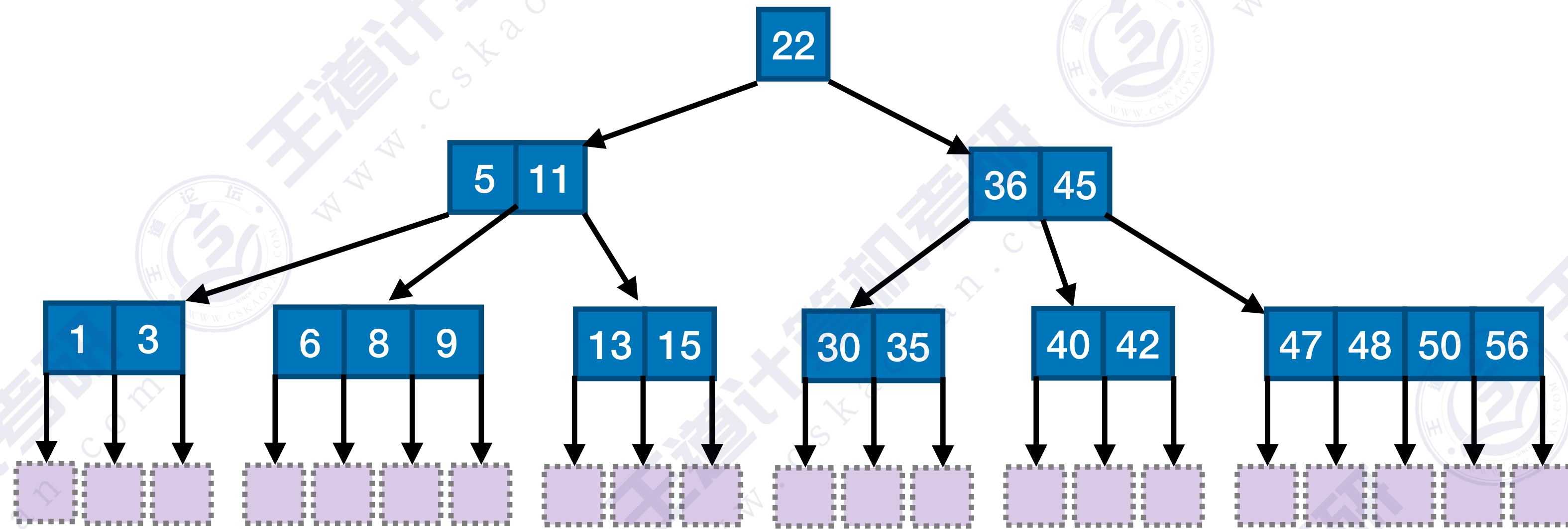
	最少结点数	最少关键字数
第一层	1	1
第二层	2	$2(k-1)$
第三层	$2k$	$2k(k-1)$
第四层	$2k^2$	$2k^2(k-1)$
...
第h层	$2k^{h-2}$	$2k^{h-2}(k-1)$

h层的m阶B树至少包含关键字总数 $1+2(k-1)(k^0+k^1+k^2+\dots+k^{h-2}) = 1+2(k^{h-1}-1)$

若关键字总数少于这个值，则高度一定小于h，因此 $n \geq 1+2(k^{h-1}-1)$

$$\text{得, } h \leq \log_k \frac{n+1}{2} + 1 = \log_{\lceil m/2 \rceil} \frac{n+1}{2} + 1$$

B树的高度



注：大部分学校算B树的**高度不包括叶子结点（失败结点）**

问题：含n个关键字的m阶B树，最小高度、最大高度是多少？

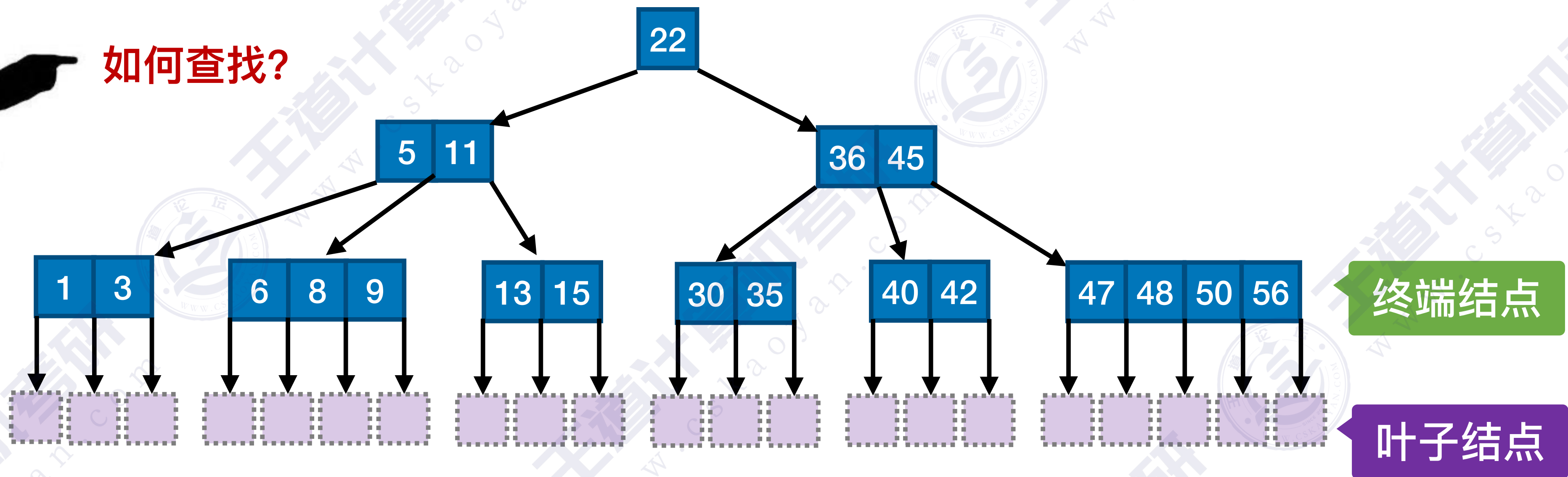
$$\log_m(n+1) \leq h \leq \log_{\lceil m/2 \rceil} \frac{n+1}{2} + 1$$

知识回顾与重要考点



字不重要，
看图！

如何查找？



m阶B树的核心特性：

尽可能“满”

1) 根节点的子树数 $\in[2, m]$ ，关键字数 $\in[1, m-1]$ 。

其他结点的子树数 $\in[\lceil m/2 \rceil, m]$ ；关键字数 $\in[\lceil m/2 \rceil - 1, m-1]$

尽可能“平衡”

2) 对任一结点，其所有子树高度都相同

3) 关键字的值：子树0<关键字1<子树1<关键字2<子树2<....（类比二叉查找树 左<中<右）

含n个关键字的m叉B树， $\log_m(n+1) \leq h \leq \log_{\lceil m/2 \rceil} \frac{n+1}{2} + 1$